Machine Learning Nanodegree Capstone: DonorsChoose.org Application Screening

Brandon M. Burroughs

March 25, 2018

Domain Background

There are many domains in which some writing is submitted and manually reviewed by a human to approve or reject the writing: scientific paper submissions to journals, book drafts to publishers, loan and credit applications, donor funding, and more. When completed by a human, many of these processes are time consuming and produce a biased result. This can give an inconsistent experience to everyone involved. However, in recent years, much has been done to improve this process in some fields, particularly loan and credit applications. According Frame et al., automatic credit scoring has led to "an 8.4 percent increase in the portfolio share of small-business loans, or $4 billion per institution." They posit that "credit scoring lowers information costs between borrowers and lenders", leading to an increased throughput of application processing. Additionally, as the credit industry has grown, a number of machine learning techniques have been tried for credit modeling, such as SVMs in research by Huang et al. using SVMs and data mining as well as research by Tomczak et.al using Restricted Boltzmann Machines.

Approving proposals for donor contributions can be seen as an analogous problem to credit scoring. Both problems are trying to best allocate funds to accomplish some goal and can be improved by automated decisioning. DonorsChoose.org allows public school teachers from around the country to request materials needed for projects and experiences for the students. These proposals can be chosen to be funded by donors. This enables teachers to provide a richer set of experiences for their kids and allows donors to fund proposals that they are passionate about. DonorsChoose.org receives hundreds of thousands of project proposals each year. These proposals are manually screened by many volunteers in a subjective and time intensive way. No sort of automation has been used to standardize or speed up the process.

Problem Statement

As DonorsChoose.org expands and becomes more popular, the manual screening process becomes less sustainable. They need to automate the screening to process more applications and focus volunteer time on the most relevant work. Using a machine learning model

to pre-screen applications, many can be automatically approved, allowing volunteers to focus on proposals that are more complicated or need further review. This also allows volunteers and other DonorsChoose.org staff focus on other areas of improvement for the overall process.

There are three primary goals to focus on.

1. How to automate manual processes to screen 500,000 proposals so they can be approved more quickly
2. How to increase the consistency of proposal screening results to provide a better experience
3. How to focus volunteer time on the most relevant areas

Building a machine learning model to predict a proposal's approval accomplishes all three of these goals in a quantifiable, measurable, and replicable way.

Datasets and Inputs

Prior teachers' proposals and the proposals' approval statuses will be used as the dataset for this project. These proposals have been previously submitted to DonorsChoose.org and either approved or rejected manually by a volunteer. This dataset provides labeled observations that look like the proposals DonorsChoose.org will see in the future. This makes it an appropriate dataset for training a supervised machine learning model to predict a proposal's approval (or rejection).

There are three primary datasets that will be used in this project:

- train.csv: This dataset provides proposal information along with a label indicating whether the proposal was approved. This dataset has 182,080 rows and 16 columns. The dataset file size is 321 MB and it uses 459 MB of memory when loaded into memory in Python. This will easily fit into memory on a standard laptop.
- test.csv: This dataset provides proposal information without a label indicating whether the proposal was approved. This is specific to the Kaggle competition that provided this data and could be used for testing and algorithm against Kaggle's held out set. This dataset has 78,035 rows and 15 columns. The dataset file size is 138 MB and it uses 196 MB of memory when loaded into memory in Python. This will easily fit into memory on a standard laptop.
- resources.csv: This dataset contains the description, quantity, and price of resources requested as part of the proposal. This dataset has 1,541,272 rows and r columns. The dataset file size is 122 MB and it uses 282 MB of memory when loaded into memory in Python. This will easily fit into memory on a standard laptop.

The train and test dataset provide many relevant data elements for predicting the proposal's approval. These include the following[1]:

- the teacher prefix: This is a categorical variable identifying the teacher's title

[1]For the full data dictionary, see the DonorsChoose Kaggle Competition Data website: https://www.kaggle.com/c/donorschoose-application-screening/data

- the number of proposals previously submitted by the teacher: This is a numeric variable
- the state the school is located in: This is categorical variable aligned to state
- when the proposal was submitted: This is a timestamp
- the category and subcategory of the proposal (e.g. "Music & The Arts" and "Visual Arts"): These are categorical variables
- the proposal title: This is a free text string
- the essays submitted with the proposal: This is a free text string

These data elements contain both raw text and structured metadata. Some of the metadata has a single value per proposal while some metadata has multiple entries per proposal. These data elements can be used in different ways as inputs to a supervised machine learning model to predict the approval status of an unseen proposal.

The class to be predicted in the `train.csv` dataset is `project_is_approved`. Approximately 85% of the projects are approved and 15% are not approved. This target is a bit imbalanced which could affect modeling. However, since this isn't a large imbalance and the dataset is sufficiently large, it shouldn't create too much of a problem. There are still 27,734 projects that weren't approved vs. 154,346 projects that were approved.

Solution Statement

To automate the proposal screening process, provide more consistent screening results, and focus volunteers' time, a supervised machine learning model can be used to predict whether the proposal will be approved or not. The dataset consisting of past proposals can be transformed and aggregated into features that will be inputs to the machine learning models. The approval status of these past projects can be used as the target to train the supervised model and let it learn the appropriate features of an approved or rejected proposal. This machine learning model will provide a more consistent screening process for the proposals and identify proposals that received low scores (i.e. the model is predicting that the proposal will be rejected) and may need more direct attention for the volunteers. There are many potential models that fit this profile: Logistic Regression with its directly interpretable coefficients, a tree-based algorithm (e.g. Random Forest or Gradient Boosting Machines) with partial dependency plots, or a more complex Deep Learning model (e.g. an RNN or sequence model) with some interpretability layer such as LIME. This machine learning model would solve the three key parts of the problem and provide a pre-screening process that could automatically approve many of the proposals.

Benchmark Model

The ideal benchmark model would be the "manual model" carried out by the volunteers manually reviewing each application. This would provide a benchmark for the ideal performance to be achieved by a machine learning model. If a machine learning model could achieve this benchmark but operate more quickly, it would solve many of the

problems of DonorsChoose.org. Since there are currently no other machine learning models or other automation being run, this would be the benchmark for comparison.

Additionally, Kaggle has provided a "Getting Started" notebook for the DonorsChoose.org competition that creates a simple model for this problem[2]. This could also be used for comparison as a benchmark model.

Evaluation Metrics

For this modeling exercise, the evaluation metric "area under the ROC curve" (commonly referred to as "area under the curve" and abbreviated "AUC") will be used. Since this is a binary classification problem, AUC provides a nice high level summary of how well the model is performing. AUC is equal to "the probability that a classifier will rank a randomly chosen positive instance higher than a randomly chosen negative one (assuming 'positive' ranks higher than 'negative')". The area under the curve is calculated by integration to calculate the area under the curve created by the true positive rate against the false positive rate[3].

Additionally, since AUC is a summary metric of the model, the Precision and Recall scores can also be used to evaluate model performance. Precision is calculated as the number of true positives divided by the number of model predicted positives. This indicates how well the model does at predicting positive instances (i.e. approved proposals). Recall is calculated as the number of true positives divided by the number of actual positives in the dataset. This indicates how well the model does at capturing all of the positive instances present in the dataset (i.e. approved proposals).

Project Design

There will be many steps involved in completing this project. This section outlines the project design and procedures.

Data Acquisition and First Checks

Since this project is part of the Kaggle DonorsChoose.org Application Screening competition, the data will be provided for download by Kaggle. Once the data is downloaded, it will need to be loaded and checked for any obvious flaws from download. Additionally, the data will be joined and viewed to ensure that everything is as expected.

[2]See the simple model at https://www.kaggle.com/skleinfeld/getting-started-with-the-donorschoose-data-set

[3]Refer to the Wikipedia page for more details on the mathematical formulation of AUC.

Data Exploration

Once the basic data quality checks have been performed, a more thorough data exploration will be performed. This includes

- inspecting each column of the data for missing values
- producing summary statistics
- producing basic plots of each variable
- plotting the variables against each other and checking correlations
- seeing how the variables against the target (project approval status)
- looking at interactions between variables and the target

First, the percent missing for each column will be calculated. This will help determine if any variables should be discarded and shed light on future imputation needs. Summary statistics produced will include minimum, first quartile, median, third quartile, maximum, mean, and standard deviation for the numeric values. For categorical values, the value counts/percentages of each variable can be calculated. As a quick pass, the most popular words and phrases can be calculated for each of the text fields and for the text fields overall. Basic histograms and bar charts can also be produced to help visualize these summary statistics.

For more complex plotting, each numeric variable can be plotted against every other numeric variable to see how the variables are related. Similar insights can be drawn from calculating correlations between each pair of variables. Boxplots can be used to see how the numeric variables vary when grouped by categorical variables. Similar analysis can be performed using the target variable to see how variables differ when grouped by the binary target variable. Finally, interaction variables can be calculated and compared against the target to see how this affects the target class.

Feature Creation

The insights from the thorough data exploration can be used to create new features from the data. While the specifics of the features will likely reveal themselves after data exploration, there are several general types of features that will be used.

- Basic features: These are features that are given directly from the data and do not need to be manipulated. One example is the number of previous proposals submitted by the teacher.
- Simple transformation features: These are features that require some simple transformation to go from raw data to feature input to the model. One example is extracting the month from the proposal submission date. Another example is creating a binary indicator for whether the teacher has submitted a project proposal previously.
- Summarizing transformation features: These are features that require summarizing information given about a proposal into a single number/category. One example is counting the number of resources requested for each proposal.
- Text features: These are features that involve using the text of the proposal submissions. These will likely be more complex and involve computing summary statistics or vectorizers over the text. They could also include variables that indicate specific

qualities or quantities present in the data such as the presence of the word "inspiration" or the number of adjectives used in the proposal. This could also include the text itself in the case of sequence models.

Model Selection and Hyperparameter Tuning

With a large variety of features created, modeling can begin[4]. With the wide variety of available models for supervised modeling, many models would be tried initially. Using a package such as `auto-sklearn` would allow for many models (e.g. Logistic Regression, Naive Bayes, SVM, Random Forest, Gradient Boosting Machine) and configurations to be explored quickly to give a general idea of what types of models work best. Since this model is being used to inform a judgment decision, a more interpretable model would likely be a better choice. This model will also be used to help focus the efforts of volunteers and insights from the model may help direct their efforts. One such interpretable model would be Logistic Regression; its direct coefficient allow for an easily interpretable prediction. Additionally, tree based models, feature importance values, and partial dependency plots allow for more understanding of the model and interpretation of each prediction.

While most of the efforts should focus on easily interpretable models, it is also worth trying more complex models as they could result in better, generalizable performance. If there is a large enough performance improvement, these models might be chosen for usage though they lack direct interpretability. Additionally, there is much research in the area of interpreting "black box" models that could be used.

Once a general class of model has been determined, tuning the model's hyperparameters can help improve the overall model performance. This would be done using cross validation to determine the best generalizable hyperparameters. This would help prevent overfitting to one specific instance of the training data.

Model Evaluation

Though much model evaluation will be happening throughout the model selection and hyperparameter tuning process, a more thorough model evaluation will occur once the model has been chosen. This will include calculating and evaluating the indicated model metrics on the held out test dataset. Additionally, this will include a more qualitative analysis of how the model performs against the stated goals of the model.

Conclusion

The application screening process at DonorsChoose.org is a great opportunity to apply machine learning. A supervised machine learning model could greatly speed up the screening process while providing a more consistent experience for the proposal submitters.

[4]While presented linearly here, feature creation, modeling, and model evaluation is often an iterative process involving all three in various orders.

Additionally, it could help direct the volunteer resources to the most relevant areas, such as proposals that are more complicated or need more assistance.

References

Frame, W. Scott, Aruna Srinivasan, and Lynn Woosley. "The effect of credit scoring on small-business lending." Journal of money, credit and banking (2001): 813-825.

Huang, Cheng-Lung, Mu-Chen Chen, and Chieh-Jen Wang. "Credit scoring with a data mining approach based on support vector machines." Expert systems with applications 33.4 (2007): 847-856.

Tomczak, Jakub M., and Maciej Zięba. "Classification Restricted Boltzmann Machine for comprehensible credit scoring model." Expert Systems with Applications 42.4 (2015): 1789-1796.

Kaggle DonorsChoose.org Application Screening: Competition Description. c2018. Kaggle Inc; [accessed 2018 Mar 25]. https://www.kaggle.com/c/donorschoose-application-screening#description

Kaggle DonorsChoose.org Application Screening: Data Description. c2018. Kaggle Inc; [accessed 2018 Mar 25]. https://www.kaggle.com/c/donorschoose-application-screening/data

Kaggle DonorsChoose.org Application Screening: Evaluation Description. c2018. Kaggle Inc; [accessed 2018 Mar 25]. https://www.kaggle.com/c/donorschoose-application-screening#evaluation

Receiver operating characteristic: Area under the curve. c2018. Wikipedia.org; [accessed 2018 Mar 25]. https://en.wikipedia.org/wiki/Receiver_operating_characteristic#Area_under_the_curve